CLAIMS

What is claimed is:

1. In combination with a computer system having a special modifiable memory in which is loaded an original code set, a method for maintaining the integrity of the contents of that modifiable memory when the system attempts to overwrite the contents with a different code set, said method comprising the steps of:

providing a one-way algorithm which acts on a replacement code set and generates a security key unique to the replacement code set, said algorithm being maintained confidential by the provider of the replacement code set;

providing the security key in combination with distributions of the replacement code set;

providing a memory controller having an embedded copy of the algorithm, said memory controller causing a tendered code set, which the computer system attempts to write into the modifiable memory, to be acted on by the embedded copy, thereby generating a local key;

comparing the local key with the security key;

allowing the contents of the modifiable memory to be overwritten only if the local key matches the security key.

2. The method of claim 1, wherein said original code set contains data and/or instructions crucial to the proper functioning of the computer system.

3. The method of claim 1, wherein the computer system also includes a microprocessor and a main memory.

4. The method of claim 3, wherein said different code set is loaded into main memory and said microprocessor executes said algorithm on said tendered code set, compares the security key to the local key, and provides the results of the comparison to the memory controller.

5. The method of claim 1, wherein said memory controller further includes an on-chip special-purpose processor and an on-chip non-modifiable memory for storing said algorithm, and access to said non-modifiable memory is limited to the special-purpose processor.

6. The method of claim 5, wherein said special-purpose processor loads said algorithm from the non-modifiable memory, calculates a local key for the tendered code set, and compares the local key with the security key.

7. The method of claim 1, wherein said algorithm employs modular arithmetic.

8. The method of claim 1, wherein said algorithm employs a cyclic redundancy check.

9. A method for preventing malicious and defective overwrites of a basic input/output system (BIOS) code of a computer system where said BIOS code is stored in modifiable memory, said method comprising the steps of:

providing a one-way algorithm which acts on a replacement BIOS code and generates a security key unique to the replacement BIOS code, said algorithm being maintained confidential by the provider of the replacement code set;

providing the security key in combination with distributions of the replacement BIOS code;

providing a memory controller for said computer system, said memory controller having an embedded copy of the algorithm, said memory controller causing any tendered code, which the computer system attempts to write into the modifiable memory, to be acted on by the embedded copy, thereby generating a local key;

comparing the local key with the security key;

allowing the contents of the modifiable memory to be overwritten with the tendered code only if the local key matches the security key.

10. The method of claim 9, wherein the computer system also includes a microprocessor and a main memory.

11. The method of claim 10, wherein said tendered code is loaded into main memory and said microprocessor executes said algorithm thereon, calculates a local key, compares the security key to the local key, and provides the results of the comparison to the memory controller.

12. The method of claim 9, wherein said memory controller further includes an on-chip special-purpose processor and an on-chip non-modifiable memory for storing said algorithm, and access to said non-modifiable memory is limited to said special-purpose processor.

13. The method of claim 12, wherein said special-purpose processor loads said algorithm from said non-modifiable memory, calculates a local key for the tendered code, and compares the local key with the security key.

14. The method of claim 9, wherein said algorithm employs modular arithmetic.

15. The method of claim 9, wherein said algorithm employs a cyclic redundancy check.

16. A method for ensuring that only an accurate copy of an authorized correct code set containing data and/or instructions crucial to the proper functioning of a computer system can be written to a modifiable memory of that computer, said method comprising the steps of:

providing a one-way algorithm that arithmetically manipulates an authorized code set to generate a security key unique to that code set, said algorithm being maintained confidential by the provider of the authorized code set;

providing the security key in combination with distributions of the authorized code set;

providing a memory controller for said computer system, said memory controller having an embedded copy of the algorithm, said memory controller causing any tendered code, which the computer system attempts to write into the modifiable memory, to be arithmetically manipulated by the embedded copy, thereby generating a local key;

comparing the local key with the security key;

allowing the contents of the modifiable memory to be overwritten with the tendered code only if the local key matches the security key.

17. The method of claim 16, wherein the computer system also includes a microprocessor and a main memory, and wherein said tendered code is loaded into said main memory and said microprocessor executes said

algorithm thereon, calculates a local key, compares the security key to the local key, and provides the results of the comparison to the memory controller.

18. The method of claim 16, wherein said memory controller further includes an on-chip special-purpose processor and an on-chip non-modifiable memory for storing said algorithm, and access to said non-modifiable memory is limited to said special-purpose processor, and wherein said special-purpose processor loads said algorithm from said non-modifiable memory, calculates a local key for the different code, and compares the local key with the security key.

19. The method of claim 16, wherein said algorithm employs modular arithmetic.

20. The method of claim 16, wherein said algorithm employs a cyclic redundancy check.